The Resource Oriented Authorization Manager (ROAM)

J. R. Burruss
General Atomics, P.O. Box 85608, San Diego, California, USA
burruss@fusion.gat.com

Abstract

The Resource Oriented Authorization Manager (ROAM) was created to provide a simple but flexible authorization system for the National Fusion Grid (FusionGrid). This system builds on and extends previous community efforts by both responding to access authorization requests and by providing a Web interface for resource management. ROAM works with the Globus Resource Allocation Manager (GRAM), and is general enough to be used by other virtual organizations that use Globus middleware or X.509/TLS authentication schemes to secure a grid of distributed resources.

Introduction

The National Fusion Collaboratory is a multi-institutional collaboration between fusion scientists and computer specialists created with the goal of advancing fusion science through more efficient use of resources and through more effective remote participation. The Collaboratory started work on FusionGrid, a computational grid, in 2001. The first resources were made grid services using the Globus Toolkit 2.0.

Early use of Globus "grid mapfiles" on FusionGrid showed that authorization was difficult because of the need for each individual host to maintain a separate grid-mapfile to map FusionGrid certificates to local accounts. Having mapped certificates to local accounts, it was left to each individual resource administrator to implement both grid-wide and local authorization policies. Experience demonstrated that it was hard to maintain coherence with mapfiles distributed across multiple machines at multiple sites. Furthermore, the use of grid mapfiles precluded variable account mapping because of the lack of wildcard specification and the need to edit each individual mapfile by hand. This is especially problematic when the same user is mapped to different local accounts on one machine depending on the resource being used, or when a resource is spread across several machines. What was needed was a flexible system to provide grid-wide authorization and account mapping.

ROAM improves upon grid mapfiles in several ways. First, it provides a coherent model for security. The ROAM information model consists of a framework of resources, permissions, users, and authorizations. Second, it allows for use-specific account mapping. Users can map to different users or groups depending on the service, an important feature when multiple services run on the same host. Third, ROAM provides administrators and users with a convenient web interface for managing authorization. Users report that ROAM is easier to use than the grid mapfiles, and administrators can quickly change authorization rules in one place as opposed to editing grid mapfiles on multiple hosts.

Information model

The center of the ROAM information model is the *resource*. A resource is typically a grid service, but it can also be an entire site, like MIT or General Atomics. In general, a resource is anything that needs to have its own authorization list. As the name suggests, ROAM is oriented to resource-level authorization control.

A *user* is any uniquely identified consumer of resources. This is typically a human with a FusionGrid certificate, but it can also be a program or service with an identifying certificate. Resource administrators and other stakeholders are users, as are FusionGrid researchers and engineers.

A *permission* is a type of usage for a resource. Each of these permission types represents a way in which a resource is used.

An *authorization* is a grant of a specific permission for a particular user on a specified resource. Authorizations are binary in nature; you either have an authorization or you do not have an authorization. An authorization indicates that user *X* has permission *Y* on resource *Z*. Authorizations may include *contexts*, which can be used to specify conditions or obligations which need to be met when exercising the permission. At the moment context is only used to specify the local user-id/group id under which an action should be performed.

Data Model

For the most part, the implementation of the ROAM information model is straightforward. Essentially, the ROAM data model is a Lampson protection matrix implemented in an object-relational database, with one table each for resources, users, permissions, and authorizations (Fig. 1). However, in order to keep the list of permissions for a given resource to a minimum, a table of *resource-permissions* was added to the data model. This table indicates the list of valid permissions for a given resource. Thus, even if new resources introduce into the database a large volume of specialized permissions, each individual resource will have associated with it only those permissions applicable to that resource.

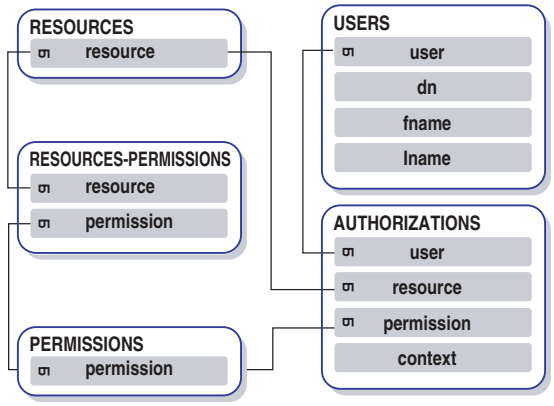

Figure 1: ROAM data model consists of *resources*, *users*, *permissions*, *authorizations,* and a table listing the valid permissions for each resource.

Architecture

ROAM has a web front-end and a database back-end. Programs communicate with ROAM through HTTP/HTTPS while users interact with ROAM through interactive web pages. Each connecting user is identified by the certificate they present to the web server. The connections are made via HTTPS with client-side authentication requested. Users without a certificate in their web browser are redirected to a login page where, assuming they enter a valid FusionGrid username and password, they are authenticated and given a proxy certificate from a MyProxy server.

ROAM avoids the push model of authorization (where the client must first contact an authorization server to get some credentials and then present them to the resource provider). Instead, clients in a ROAM-enabled grid connect to resources as they would normally, using an X.509 proxy credential from MyProxy to authenticate (Fig. 2). The resource then consults ROAM to see if the connecting user is authorized. Note that under ROAM, it is ultimately the resource that is empowered to make policy decisions.

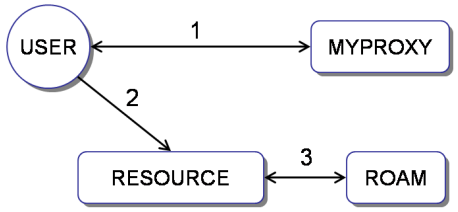

Figure 2: A users first signs on to FusionGrid using MyProxy, then accesses a resource directly; the resource then checks ROAM for authorization

Conclusion

The first usage of ROAM was to represent the authorization policy for the GATO FusionGrid service running on computers located at General Atomics (GA) in San Diego. To use the GATO service, users must be authorized by the code's authors and administrators to use the code, *and* satisfy the GA site security staff by signing certain data access and network usage agreements. This authorization was modeled in ROAM by creating a resource called GA with a permission called access, and a resource named GATO with a permission called execute. To use the GATO service, a user must have both of these authorizations. In this way, either stakeholder — the site security staff at GA or the administrators of the GATO code—can control usage of the GATO service.

ROAM has satisfied the different classes of stakeholders: FusionGrid developers have abandoned grid mapfiles in favor of ROAM for secure MDSplus databases at the DIII-D and Alcator C-Mod fusion facilities, administrators have used ROAM to control access to their resources, and fusion scientists have used the ROAM interface to request resource authorization. FusionGrid developers plan to use ROAM with future services and to add ROAM to the existing TRANSP service to test with two orders of magnitude more users.

Acknowledgments

This work was funded by the SciDAC project, US Department of Energy under contract number DE-FG02-01ER25455 and cooperative agreement number DE-FC02-04ER54698, and by the Director, Office of Science, Office of Advanced Science, Mathematical, Information and Computation Sciences of the US Department of Energy under contract number DE-AC03-76SF00098.